Preservation strategies for an internet-based artwork yesterday, today and tomorrow

Subtitle

Claudia Roeck
University of Amsterdam
The Netherlands
c.rock@uva.nl
ORCID

Rafael Gieschke
University of Freiburg
Germany
rafael.gieschke@rz.uni-freiburg.de
ORCID https://orcid.org/0000-0002-2778-4218

Klaus Rechert
University of Freiburg
Germany
klaus.rechert@rz.uni-freiburg.de
ORCID

Julia Noordegraaf
University of Amsterdam
The Netherlands
j.j.noordegraaf@uva.nl
ORCID

Abstract This paper investigates possible preservation strategies for an internet-based artwork and assesses the strategies that best capture the authenticity of the work for future iterations. Two different preservation strategies are applied for the internet-based artwork TraceNoizer.org from 2001. A third one, a Linux Live CD, was carried out by one of the artists. They are compared and evaluated from the perspective of the long-term preservation of the work's most significant properties. Compared to software-based artworks, the characteristics of internet-based artworks shift the focus of the preservation measures from the stabilization of the software to reduction of server maintenance, protection of server and artwork from internet threats and reduction of external dependencies. This paper suggests solutions how to handle these challenges and discusses its benefits and disadvantages for long-term preservation.

Keywords internet-based art, software-based art, sustainability, long-term preservation, preservation strategies

Conference Topics Designing and Delivering Sustainable Digital Preservation

I. Introduction

Until a few years ago, Internet-based art was not widely collected in contemporary art museums and collecting institutions. The Guggenheim Museum in New York was one of the first contemporary art museums to acquire web-based art (net.flag acquired in 2002). Rhizome, an art organization in New York, has probably the widest experience in the preservation of Internet art. In contrast to most museums, they are focusing on purely digital art. Their collection consists of several hundred digital artworks. The number of museums collecting internet-based art is slowly increasing. For instance, the Stedelijk Museum Amsterdam (NL) acquired several web-based artworks jointly with the MOTI museum in Breda (NL) in 2017, LIMA in Amsterdam is hosting web-based artworks by and for Constant Dullaart since 2018 and the house of Electronic Arts in Basel (CH) acquired about 20 web-based artworks between 2016 and 2018. While the museums slowly start to acquire Internet art, the preservation, change management and hosting of these artworks is often not solved. This is the reason why an internet-based artwork was chosen as a case study for this research. Its embedding in the Internet is a specific feature relevant for its preservation that does not exist for software-based art. It makes the work vulnerable towards changes of the internet environment. While it is logical to assume that software-based art subsumes internet-based art, the term will be used more narrowly in this article: it will exclude internet-based art, in order to be able to differentiate between them.

This article shows different ways for dealing with the long-term preservation challenges of an internet-based work. They are demonstrated on a case study, *TraceNoizer* (2001) by LAN, acquired by the House of Electronic Arts Basel (HeK) in 2017. After defining its significant properties, different digital preservation strategies are applied, their sustainability and their impact on the authenticity of the artwork examined and compared. One of the artists carried out a preservation strategy already in 2004, which offers an interesting opportunity to study its long-term effects. Finally, the differences between the preservation strategies for software-based and internet-based artworks will be discussed.

16th International Conference on Digital Preservation
iPRES 2019, Amsterdam, The Netherlands.

DOI: 10.1145/nnnnnnn.nnnnnnn

II. Related Works and Definitions

Dekker in her dissertation describes the performative nature of such works. In her view, because of the great variability in their appearance and behaviour, it is not possible to conserve an actual net artwork. While we also find that it is not possible to reconstruct the actual, original appearance and functionality of *TraceNoizer*, we do propose strategies for preserving the significant properties of this work for future iterations.

Phillips et al. report the restoration of Shu Lea Cheang's Early web-based artwork, Brandon (1998-1999), at the Guggenheim Museum in [2] and [3]. They migrated the work (about 65'000 lines of code) to a current Web server and annotated the changes they had to carry out in order to make it compatible with today's Web browsers. This huge effort was done with the help of computer science students. Future migrations will necessitate repeated efforts and introduce changes with each migration. Not many institutions have the means to do repeated migrations every few years. Hence, this paper will compare different preservation strategies in order to find more sustainable solutions.

Miksa, Mayer, and Rauber are proposing strategies for businesses whose processes depend on Web services. They are suggesting to create "mock-ups" for these Web services. These mock-ups do not actually process requests but instead pull the response from a database of the mock-up. For this purpose they recorded the request and response streams of this Web service. Espenschied and Rechert suggested this "mock-up" strategy for "Apparently Infinite Objects", in particular internet-based artworks. Implementation, its feasibility and efficiency, remain open. Besides, Espenschied and Rechert proposed a stub interface and the mirroring of Web services to deal with external dependencies, that will be discussed in section VII and VIII of this paper.

Web archiving can be applied to preserve certain or parts of internet-based artworks. In 2014, Mat Kelly highlighted in a presentation [6], that a Web crawler usually changes the capture context of the Web browser. As a consequence he formulates high level requirements for the creation of Web archive files (WARC). In particular, he asked for a crawling software to capture the embedded scripts of a Web page and to allow the user to execute these WARC files in an Web browser. Rhizome subsequently developed such a system with the webrecorder and webrecorder-player [7]. These approaches, however, are only capable of capturing the "surface", for a truly internet-based artwork and thus have to be extended with solutions that also preserve its logic.

For this research, we are building upon Roeck, Rechert, and Noordegraaf using the same method (determine significant properties, compare different strategies and evaluate based on criteria for long-term preservation). However, the analysis was not geared towards internet-based artworks which is why they deserve further attention.

III. Significant Properties and Dependencies of *TraceNoizer*

A. Starting from a digital ruin

TraceNoizer is an interactive website created by the art collective LAN (Local Area Network: Annina Rüst, Fabian Thommen, Roman Abt, Silvan Zurbruegg and Marc Lee) in 2001. Anybody who had access to a computer with a screen, keyboard, mouse and Internet connection could experience it. The House of Electronic Arts Basel acquired the work in 2017 as a pile of code. *TraceNoizer* had not been online for many years. Even though the source code was available, it was not fully functional.

As it is common for institutions to acquire works a few years after their creation, many websites such as *TraceNoizer* have to be reconstructed without having a functional reference. The project archive delivered by the artist is in itself a blurry object, insofar, as its past significant properties are not precisely known and the code that is relevant for the work not clearly delimited. The plethora of versions and customized presentations, adaptations to free Web services and APIs, improvements of functionality and Web design, multiple backups, handwritten maintenance programs and the simultaneous lack of documentation turns the archive into a maze for a conservator or curator. Even though the artwork itself might not have been blurry, its data, which is the base of reconstruction, might be. Hence, the process of the definition of significant properties is split up in two steps:

1) The work will be described as a reconstruction of the past (what we think it worked and looked like in 2004. The 2004 version is the most recent version)

2) Based on that, the significant properties of the artwork restoration will be determined (transfer to the present and future).

As this process and its results are subjective, the decisions will be supported by reasons so that others can understand the motivation to determine certain properties as significant. Hence, "significant for restoration" means "considered significant for restoration".

B. Idea of the work

In the beginning of the 2000s, the aspect of data autonomy was widely discussed among critical Internet users. The fact that it was often impossible to delete one's own traces in the Internet motivated LAN to create *TraceNoizer*. The work was inspired by the "Jam Echelon Day" on the 21 of October 1999, when the international hacker scene decided to flood surveillance agencies such as CIA with fake information. LAN applied a similar counter strategy with *TraceNoizer* in order to diffuse the traces of one's own personal data on the Internet. On the *TraceNoizer* website the user could enter his/her name. Subsequently, the *clone engine* searched for all the websites containing this name and reassembled a new

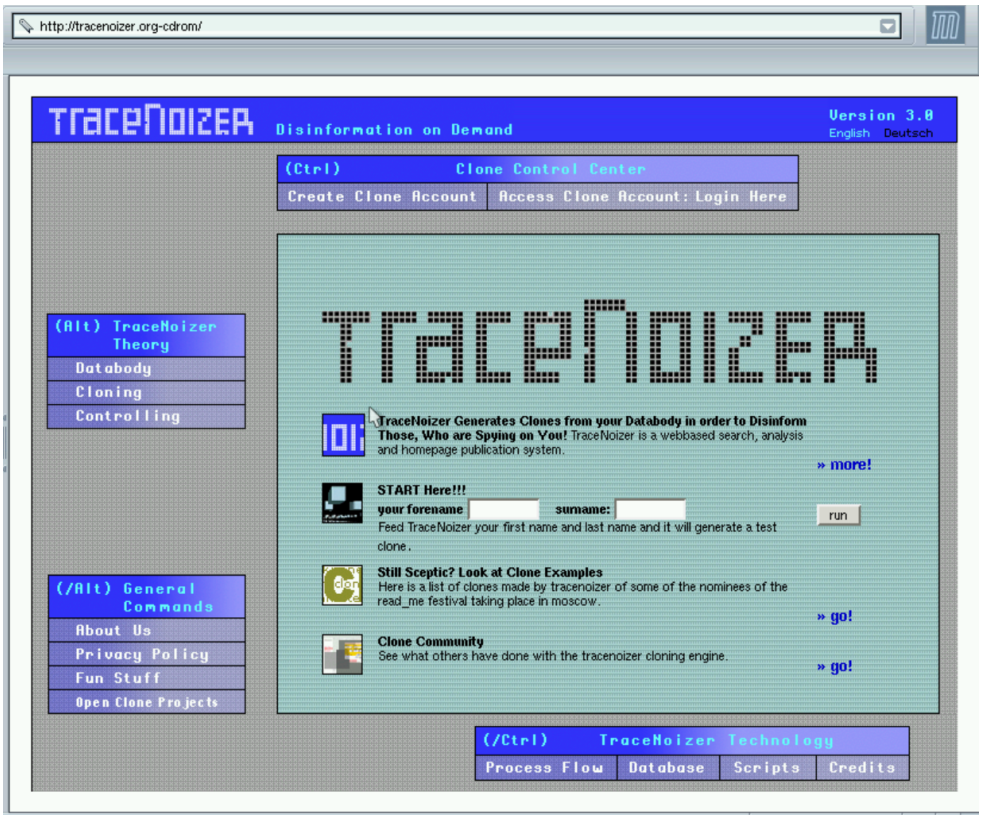

Figure 1: *TraceNoizer* by LAN. Screenshot Knoppix CD

personal website out of these search results by using an algorithm that follows a similar logic as the ranking of the search machines. Thus, theoretically, in a following search, the search engine should rank the generated website higher than the original ones. The generated websites (so called clones) were uploaded to free website hosting platforms that indexed these new clones so that there was a chance for the search machine to find them. The more this process was repeated, the higher up climbed the ranking of the clones until the "original" pages did not appear in the search results anymore.

The artists highlighted in an interview, that *TraceNoizer* was a performative tool and not a static website. According to the same interview, another important aspect of the work was the automatic generation of websites without human interaction except for entering their name.

The artists called the generated websites clones, which is why this article continues to use the term clone, although they are not clones in a literal sense, but assemblies of text and images from other websites. The programs that produce the clone are subsumed under the term *clone engine*.

Significant for the restoration The cloning process was supposed to be repeated until the clones themselves appeared in the search results and were used to produce the new clones. However, there were some discussions among the artists, whether the high search engine ranking of the generated clones really worked. This doubt is confirmed by jury members of the readme festival (2002) where *TraceNoizer* was exhibited. According to their experiences their clones would not appear in the search results. Hence, the original idea was not perfectly executed in the 2004 manifestation of the work.

C. *Processes implemented in software version 2004*

The *TraceNoizer* logic consists of three main parts: the graphical design of the website, the *clone engine* where the clones are generated and uploaded to the Internet and the *clone control center* where the user could access his/her clones. The graphical design will be discussed in the sections D "Look and Feel" and F "External Dependencies". The trace_centralV4.pl script represents the most important script of the *clone engine* as it creates the clones. Another central element is the database that serves as a temporary data storage for the clone generation and a permanent storage for the user logins and the location of the uploaded clones. After the user entered the search term (the user's name, but other names can be entered as well), the *clone engine* started to search the Internet via a search engine interface. Up to 40 search results (websites) were saved in the database. Their text was extracted, temporarily saved in text files, then the text was split up in sentences that are saved in the database. The text was statistically analyzed by using the rainbow library [1]. The ten most frequently used words were saved as keywords in the database. The texts and sentences were ranked according to the frequency of the

[1]Information about the rainbow library: https://www.cs.cmu.edu/ mccallum/bow/

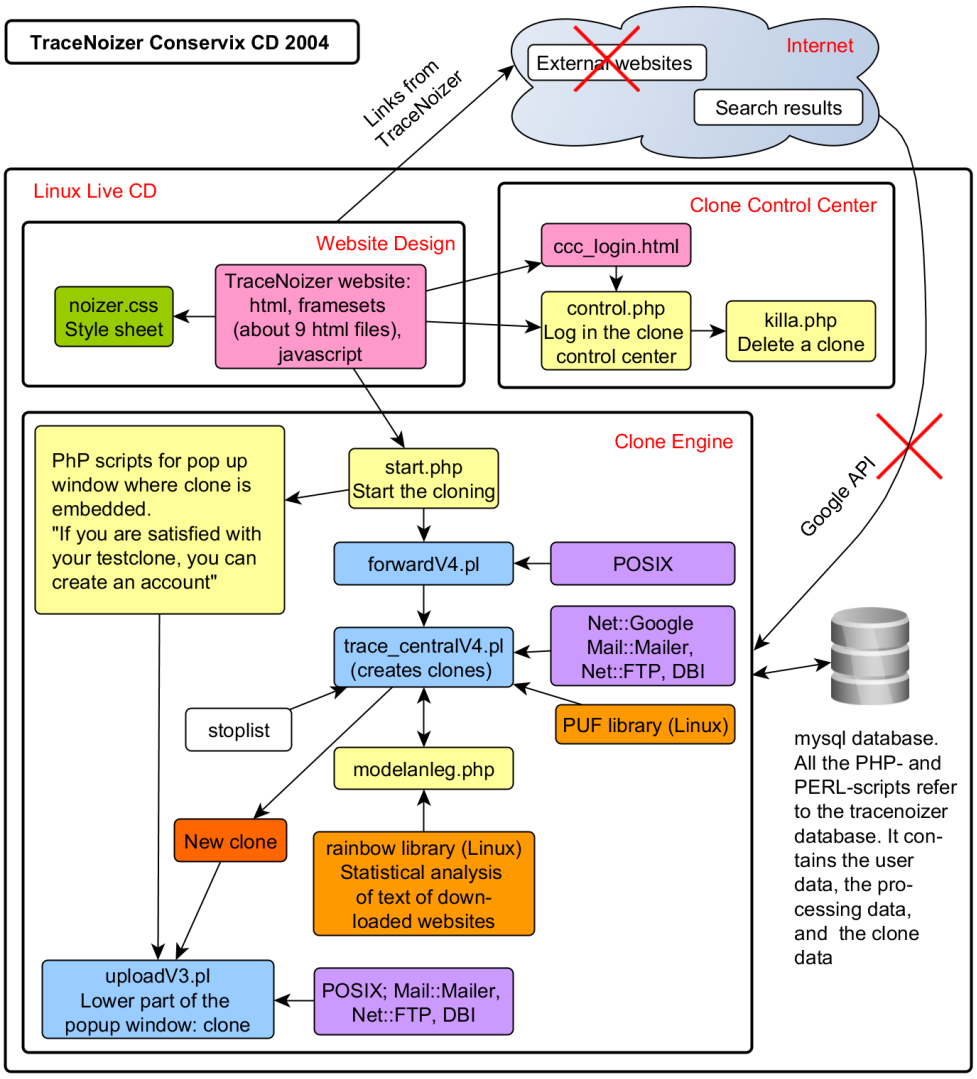

Figure 2: *TraceNoizer* in the Linux Live CD version

keywords. This was the basis for the generation of the clones that mimic personal websites of the previously entered name. Each clone had a main page with ten sub-pages, each of these sub-pages corresponding to a keyword. Each keyword page consisted of an image, a text and an external link. The clone was then uploaded to a free host and the temporary database entries and text files were deleted. If the user was not satisfied with the created clone, he/she could log in the *clone control center* and delete the clone from the host. The user login could be different from the name entered in the search. However, each user could only manage one clone.

Significant for the restoration The scripts and programming languages are the material of the website. The way the scripts are programmed is typical for that time period and hence considered as significant for preservation.

D. Look and Feel: Graphical Website Design

The look and feel of the *TraceNoizer* website can be investigated by running the Conservix CD with Mozilla Firefox 1.3, released in 2002. There are no requirements for the Web browser such as specific plugins. The monitor resolution common at this time was 1024 x 768 pixels.

The graphical design consists of a puzzle of website pieces (frames) that are hold together by the index.html-page and are re-used for the different sub-pages. This feature is not supported in HTML 5 anymore. The background of the links change color, if the mouse hovers over it. This is achieved by JavaScript. While the website design is very clear, logical and functional, from a present-day perspective the design looks a bit outdated.

Not only the *TraceNoizer* website needs to be evaluated, but also the generated clones. The clones are built much simpler (s. figure 3). They just consist of a title, a centered image and sub-pages. The sub-pages also have a title, a centered text and an external link on the bottom. Except for the images they do not contain any graphical elements. Even for 2001 standards the aesthetics of the clones are crude.

Significant for the restoration The look and feel of the *TraceNoizer* website is considered as a significant

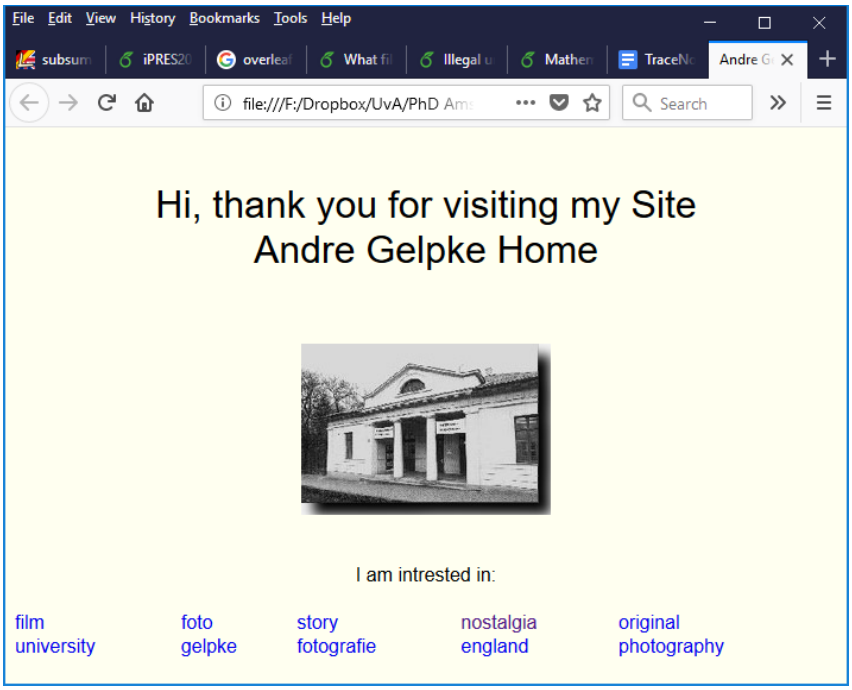

Figure 3: clone from the clone-it project 2001 in Firefox 60 (2018)

property as it points to the time of its creation. As a second priority, the framesets should be preserved, as a typical technology for that specific period.

As the clones had already looked quite crude in 2004 (They just consisted of text, images and links), there is no reason to change the clone design today. Hence, the clones generated with the *clone engine* today should look the same as they looked back then.

E. Environment

TraceNoizer contains external links that reach into the World Wide Web. Other artists such as Darko Fritz or Knowbotic used the *clone engine* to create their own artworks. The links to both artworks are broken. LAN themselves used the *clone engine* for two events in 2001 and produced a plethora of clones. These clones are not online anymore, but the artists have stored them on the Conservix CD.

Significant for the restoration The above mentioned Web pages of related artwork projects are considered significant, as *TraceNoizer* links to them directly and the artists used the *clone engine* to create these projects. Hence, these Web pages should be preserved as context of *TraceNoizer*. It could even be argued, that the indirect environment such as the artist's websites, Web pages from *TraceNoizer* exhibitions and the Echelon-Web page should be preserved as context, which is a second priority. This context demonstrates the performativity and project character of TraceNoizer.

F. External Dependencies

Dependencies that reach outside the artwork's Web server to Web servers that the artist and the collecting institution have no control over are typical for internet-based artworks. For *TraceNoizer*, this applies to the use of the Google search engine. The PERL interface to the Google search engine used for this work has been obsolete since many years and does not work anymore. The upload of the clones to free hosting platforms is another fragile dependency, as the free hosting platforms and their requirements to host pages for free change often. The artists experienced that too and asked friends for free server space to store the clones. For this reason the FTP clone upload was deactivated in the 2004 version of the code. Finally *TraceNoizer* depends on the HTML code of websites of the period around 2004. The *clone engine* extracts text, images and links from websites in order to reassemble them later. If this extraction does not work well, the newly assembled clones will be faulty. However, there must have been a certain percentage of faulty clones already in 2004 due to the automatic clone generation.

Significant for the restoration The different versions of the *clone engine* show that the artists switched from Google search engine in 2001 to Yahoo in 2002 and back to Google in 2004. The Google search engine itself changed considerably since 2004[2]. How the changes affect the artwork is not clear, but the artists did not seem concerned about it. In order to be consistent with the creation period a search engine from 2004 would be preferable to a current one. Knowing, that this is not possible, any search engine based on the page rank algorithm[3] is acceptable.

[2] "With some 1600 improvements to Google Search in 2016 alone, these are just a sample of some of the ways we have been making Search better and better over time." [9]

[3] https://en.wikipedia.org/wiki/PageRank accessed 2019/11/02

Due to the fact, that LAN stopped uploading the clones to free hosting platforms, the accessibility of the clones in the Internet, but not the use of free hosting services is declared as significant.

The change from web1.0 to web2.0 affects the look of the clones. Nowadays, the html code is not "hand written" as it was often the case in 2001, when personal websites were more common. Today, a name will rather appear in a social media platform whose html code is automatically generated. As the legibility of the clones is significant for the understanding of the work it would be considered satisfying, if most clones had the sub-pages corresponding to the most frequent keywords and if they had working image links.

G. *Conclusions*

Due to nature of internet-based artworks, dependencies on external services and external websites that are part of the ever evolving network are a real challenge. These external dependencies can be split up in technical dependencies and content dependencies. Espenschied's technical definition of a blurry object in [5] refers to technical external dependencies. Such external technical dependencies (such as the Google search engine in *TraceNoizer*) cause variations in their execution, as these dependencies change. In software-based artworks, similar variations are caused by replacement of hardware such as computer and peripherals.

The blurriness of the object extends beyond its technical external dependencies, in that internet-based artworks also have content-related dependencies with external objects/sources. As it is usually not possible to confine a context of a net-based artwork, its delineation is often subjective or dependent on circumstances. Strictly speaking one could argue that the clone-it project and the links to external websites or related art projects is not part of the work, but in order to fully understand the purpose and effect of *TraceNoizer* they become part of the work. This kind of interlocking with the environment is typical for internet-based artworks.

The performativity of *TraceNoizer* consists of the clone generation (*clone engine*) and the clone management (*clone control center*) by the user. The fact that the input is generated by search results from the Internet poses challenges for preservation, as the properties of the Internet are gradually changing.

Finally, the user management and databases are a typical property of networks. New user specific data has to be saved and becomes part of the work. Although user specific logins might also be part of certain off-line artworks, it is much less common and it is limited to the visitors of the physical artwork.

IV. Long-term preservation criteria

In order to find and evaluate preservation strategies for the above mentioned properties typical for internet-based artworks, the following criteria for sustainable preservation strategies are suggested. Criteria for assessing the long-term sustainability of preservation measures were established for software-based art in [8] and are hereby assessed for web-based artworks.

Adaptability to new hardware is relevant but not crucial for the back-end. Web servers are generic computer hardware without specific components such as super fast video cards or specific input or output devices. Almost the same can be said about client computers. In order to view the artworks, they do not use any specific hardware features except for certain generic input devices.

The *ability to deal with software obsolescence and changed network protocols* is very relevant, as the software of an internet-based work is usually updated when transferred to a newer Web server. In addition, the work needs to adapt to new network protocols that are updated periodically (for instance HTML every few years). The change of application programming interfaces (API) is another frequent cause for malfunction internet-based artworks and an example for software obsolescence.

The *stabilization of software complexity and the minimizing of the software change rate* is very relevant for web-based artworks, as they are subject to fast and frequent changes due to the Internet connection. As their environment changes fast, the risk is high that such works quickly become outdated if not dysfunctional. In order to prevent that, they need to adapt, too.

The *ease of installation* of the artwork and of connecting peripherals is not relevant for web-based artworks.

The *reduction of maintenance* is very important for internet-based artworks, as maintenance of internet-based artworks, especially of *server-side dynamic websites*, can be laborious. Tasks such as the detection and cleaning of abusive and resource-intensive processes, updating the server software, and maintaining the database and server health can take up to one hour per week, which, in a museum context, is quit intense.

The *scalability of a preservation strategy* is relevant for web-based artworks, depending on the number of artworks an institution hosts. The more artworks, the more methodical the hosting and preservation approach needs to be in order to be able to exploit synergies and to reduce the maintenance per artwork.

V. Linux Live CD (Conservix CD)

In 2003, Fabian Thommen produced a CD-ROM with a bootable live operating system based on the knoppix-technology that he named Conservix. Conservix is set up with a basic Linux operating system, an Apache Web server, a MySQL database, the programming languages PHP and PERL and the Web browser Mozilla Firefox 1.3[4]. With this live system, the user did not need to install

[4] Operating system Debian 1.3-4, Web server Apache 1.3.27, database MySQL version 11.18, distribution 3.23, programming languages PHP (4.1) and PERL 5.8, and Web browser Mozilla Firefox 1.3

anything or change computer configurations. The computer starts from the CD in place of its own boot system. It automatically opens *TraceNoizer* that is installed as a dynamic website (s. figure 1).

The Conservix CD fulfills the previously defined **significant properties** partly. It was possible to temporarily create clones with it, although they could not be uploaded to external Web hosting services, and the database entries could not be permanently stored as the CD is read only. For this reason, the user could not manage the clones in the *clone control center*, partly diminishing the performative nature of the work. The client computer and the Web server coincide in one machine. An important disadvantage of the CD is, that it cannot be directly accessed through an URL, but needs to be installed in an emulator in order to run. Due to the obsolescence of the search engine interface, the CD-ROM does not allow to generate clones today, while the website as a graphical interface is displayed without errors. The CD-ROM also comprised the clones of the clone-it project so that it is known today, what the output of the *clone engine* looked like. Unfortunately, the input stream (search results) of the *clone engine* was not recorded, so that it is not possible to verify today, whether a restored version would produce the same output. Other "damages" are external links on the *TraceNoizer* website that are broken in the meantime.

Regarding the **sustainability** of the Conservix CD from 2003 it can be stated, that its iso-image still exists in 2019 and that an Intel or AMD-compatible processor can still run it. The Conservix CD is *able to deal with software obsolescence and changed network protocols* such as HTML as it contains both server and client. When it comes to external dependencies such as the Google API, Conservix does not cope so well. At least, it would be convenient if the CD produced an error message saying that the Google API does not work anymore. The Conservix CD *stabilizes software complexity* as it is read only. The *Maintenance* is also low, but would include the periodical updating of external dependencies such as Google API and thus producing a new CD-ROM. The security risks are zero for the host computer, as the CD is read only and the computer does not use its own operating system. The preservation strategy is *scalable* in so far, as Linux Live CDs can be produced for other internet-based artworks.

The Conservix CD played an important role in the definition of the significant properties: It documents, how the work is installed, what libraries and program versions, and what browser were used. It recreates the look and feel of the work without having to install much, except for a generic emulator. On the other hand, a Linux Live CD is not the best solution for internet-based works with external dependencies such as a Google search API. It also gives a false sense of security, as Fabian Thommen mentioned that he did not install the work exactly the same way as on the Web server. For instance, administration programs to maintain the database are not necessary on a CD ROM.

VI. Preservation version "Migration"

This section describes the preservation measures undertaken by Fabian Thommen in 2018. He migrated the work to a current Web server with an old PHP version 5.5.9 and made the following changes:

- He replaced the Google library from 2004 with a Google Custom Search API from 2018.

- The database commands in PHP had to be replaced in order to be compatible with newer PHP versions.

- Configurations like the database connection and the Google API keys were moved to a configuration file in order to reduce maintenance and increase security.

- Security was enhanced in parts. One big security risk is the passing of variables such as user data from the client to the server. There are different methods, how a browser client can send information to the Web server. In *TraceNoizer* the user variables were passed as register_globals to the PHP script. This method is insecure, as the input can be easily manipulated. Thus, from PHP 5.4.0. on only the GET and POST methods are possible. Fabian Thommen adapted the scripts accordingly.

- *TraceNoizer* used the rainbow library in order to analyze the text of the websites generated by the user's search. The latest rainbow version dates from 2002 and its binary was compiled for a 32bit operating system. To run the 32 bit binary on a 64 bit operating system, the library lib32z1 had to be installed.

The **significant properties** of the work were partly preserved: With this migrated version, clones can be generated and the user can login and delete his/her clone. The scripts and programming languages are only so much changed that they function. The look and feel of the website stays the same, even if the website is viewed on a current browser. This will not be the case in the future, as frames are not supported in HTML5. However, this can be solved with a browser emulation. Almost all the significant properties are respected with one exception: the clones are faulty. They are so faulty, that they do not fulfill the purpose of pretending to be somebody's homepage (selection of keywords, missing images, s. figure ..). Most of them do not contain images and the sub-pages made of keywords are also missing. This can be caused by malfunctioning of the rainbow library, or the fact that the structure of Web pages has changed so drastically since 2004, that the extraction of sentences and images does not work properly. It is also possible that the *clone engine* has never functioned as intended and has always produced a certain amount of faulty clones.

Regarding **sustainability**, the migration strategy does not yield the best results. It is able to deal with *software obsolescence and changed network protocols* such as HTML or the Google API, but at the expense of changes in the code. In addition, these changes have to be re-

peated every few years to keep up with changes in Web technology. Hence migration does not stabilize software complexity but rather enhances it. *Maintenance* will be high in order to alleviate security risks but also to clean up the database periodically. The *scalability of the preservation strategy* is relevant for the House of Electronic Arts, as they host other internet-based artworks. However, as each works needs an individual Web server software environment, the migration strategy for this work does not scale.

It can be summarized, that the migration strategy met the significant properties of the *TraceNoizer* website, but not the ones of the clones. It's biggest shortcoming from the perspective of long-term preservation is the fact that it needs to be repeated regularly. Serious Internet security concerns and an expected high amount of maintenance add to the disadvantages of this strategy.

VII. Preservation version "Emulation"

As an alternative, another preservation strategy was tried by emulating the Conservix CD of *TraceNoizer* using the University of Freiburg's *Emulation-as-a-Service* (EaaS) framework[5]. EaaS provides users with convenient access to emulators via their Web browsers. A curator can ingest a digital object, configure the right emulator and its settings, and allow any user to start the configured emulation environment. By default, each user is presented with a fresh emulation environment as configured by the curator and several users can use different sessions of the same emulation environments at the same time. Alternatively, however, emulation environments can also be "snapshotted" by users or curators, conserving their current state including any manipulations by the user.

Previously, EaaS concentrated on emulating single (unconnected) environments, e.g., a single preserved work of digital art as an archived CD-ROM image, which needs to be run on a Windows-95 environment with an installed Macromedia Flash player. As is outlined in this work, preserving single works as stand-alone entities is not sufficient for many works as their significant properties are realized through and depend on the combination of several systems. Thus, it is necessary to regard the whole ecosystem as one (connected) preservation environment.

To facilitate the emulation of a connected environment, a recent addition to EaaS allows to create a virtual network, which operates on the Ethernet layer [10]. The virtual network is represented by a URL for identification and for access control. Via this URL, the virtual network can be reached from the Internet using the WebSocket protocol over HTTPS. HTTPS/TLS encrypts the traffic and, thus, shields it from malicious access from the Internet, while the usage of the WebSocket protocol (as opposed to direct TCP/IP) shields the Internet from the emulated environments (which malicious users of the EaaS system might otherwise abuse to perform, e.g., DDoS or spam attacks on the public Internet). At the same time, it introduces a layer of emulated Ethernet traffic on top of the EaaS Web API and, thus, shields the EaaS host system from Ethernet traffic between the emulated environments. Inside the WebSocket connection, Ethernet frames from the connected emulation environments are prefixed with a simple two-octet big-endian length header (the same format as used by the VDE 2 library[6]).

Firstly, the described concept allows to connect multiple emulation environments. The emulation environments can either be specific to the preserved digital object (e.g., in a multi-machine system consisting of an application server and a database server) or generic emulation environments can be combined ad-hoc. This approach allows to easily reuse emulation environments by a curator or a user without any necessary special knowledge.

In the case of *TraceNoizer*, its Conservix-CD version was emulated by the EaaS framework using the *QEMU* emulator (s. figure 4, Option 1). In a second connected emulation environment, a contemporary Web browser was started. As a further step, other emulated environments containing Web browsers could be built to allow users to examine *TraceNoizer* (and any other digital objects). As *TraceNoizer* was originally built in a time in which optimizing Web sites for specific Web browsers (and build upon their non-standardized features) was prevalent, this could be essential to fully reproduce the original performance of the artwork as perceived by different users at that time.

Secondly, the approach of virtual networks allows to offer additional services in the network. E.g., it is currently already possible in EaaS to connect the virtual network to the current live Internet, and thus, allow a connected user to access current Web sites. To fully recreate the original *TraceNoizer* performance, instead of allowing access to the live Web, an archived version as, e.g., provided by the Internet Archive could be used (s. figure 4, Option 2). This would effectively operate as a transparent proxy[7], operating at either the DNS and/or HTTP layer. The proxy could either be preset by a curator or configured by the user to serve the Web as archived on a specific date. This would allow, e.g., to retroactively analyze the behavior of *TraceNoizer* at different points in time.

The virtual network cannot only connect emulated environments but also allows (via *eaas-proxy*) applications from outside to access the services provided in the virtual network. It, therefore, allows to map and forward an external TCP port to an internal IP address and TCP port. The termination of this connection can either occur in the public Internet (at an individually assigned network port for each user session), at the user's computer on localhost (by downloading and running a eaas-proxy binary), or, in the future, directly in the user's

[5] https://gitlab.com/emulation-as-a-service

[6] https://github.com/virtualsquare/vde-2

[7] Also known as interception proxy, see RFC 3040, section 2.5, https://www.rfc-editor.org/info/rfc3040

browser (using ServiceWorkers [11]). It allows the user to view *TraceNoizer* in their current Web browser, "breaking out" of the emulated environment and makes operation like interacting with the website content, copying contents, or even deep-linking to contents much easier. The user is, though, still protected from any malicious emulation environments by their Web browser, and the operator's system is protected by the virtual network, which is directly connected to the emulated environment but separated from the actual host system.

A further problem, which can be solved by the presented approach, is the usage of ancient Google search Web APIs (utilizing SOAP) for the *TraceNoizer* system. Google has stopped supporting this API, but it can be emulated in a virtual network environment and, consequently, allows the *TraceNoizer* environment to remain unchanged. The same approach is applicable for *TraceNoizer*'s storing clones on Web hosts which have long ceased operation.

VIII. Conclusions for the preservation strategies for internet-based artworks

For internet-based artworks, that originate in the fast-changing technical and social environment of the internet, it is necessary to abstract from the concrete technical setup and formulate more high level principles. The following three principles can be used as guidelines for maintaining the functionality of internet-based artworks in a new networked environment:

a) Simulation: Part of the old environment is recreated in order to adapt to the old artwork and allow better interaction without having to change the original artwork too much.

b) Bridging: A bridge can be built between the old artwork and the new environment. This can be achieved by encapsulating the artwork and then providing an interface to translate the input/output between the old and the new environment. It could also be achieved by adapting the code of the artwork directly to communicate with the new environment, which would correspond to a migration.

c) Reinterpretation / Reconstruction: The artwork, or parts of it, could be recreated by different means, such as new platforms or new technology, in order to adapt to the new environment. For instance *TraceNoizer* could be recreated on the Facebook platform, by cloning Facebook accounts instead of websites.

Web archiving is an example for simulation. Broken external links of *TraceNoizer* could be downloaded from the Internet Archive and saved in a protected environment. This protected environment is stable and in control of the conservator.

The emulation of the Google Search Web API as described in the previous chapter is an example of the bridging strategy, as it allows to use the Google Search Engine. The exchange of the Google Search API with the current API as done in the migration of *TraceNoizer* is also seen as bridging from the "old" work to the new environment.

The above mentioned principles are carried out by applying a combination of the well-known digital preservation strategies reprogramming, migration, emulation / virtualization and web archiving. As the examples showed, these strategies and principles can be applied to different elements of the website such as web browser, web server, or to its external dependencies such as parts of the world wide web environment and external web services.

External web services are used more and more in internet-based artworks. External web-services can be preserved or handled as described by Dragan Espenschied and Klaus Rechert in "Fencing Apparently Infinite Objects". They are suggesting the mirroring of the web-service, a stub interface that has a reduced functionality instead of the web service, and the recording of network traffic of a web service for a limited number of queries. All three proposals correspond to the principle of simulation mentioned above. If simulation is used as a strategy to preserve web services instead of bridging with an API, it is likely that a compromise with the functionality or authenticity of the work has to be accepted.

In order to link different emulated elements of the internet-based artwork, the University of Freiburg enhanced EaaS by enabling to connect these emulated environments within a virtual network. This allows a great flexibility in finding tailor-made preservation solutions. In addition, the user can login in this network and save his/her data by taking emulator snapshots. This has the advantage that the Web server including database can be reset for maintenance purposes without losing user data.

Software-based art relies on the same digital preservation strategies as internet-based art with the exception of web-archiving. However, for software-based art, hardware can play a much more important role than for internet-based art. Hardware can be compared to an external dependency, as specific peripherals cannot be easily rebuilt and their production is dependent on the production company. Hence, the bridging principle is important when replacing old equipment with new equipment. Emulation of input/output devices is an example for this. The simulation principle would be applied when running a new piece of equipment within a case of an old device.

The networked environment in an internet-based artwork does not exist for software-based artworks. The latter rather has a physical environment that is often variable and might be determining for the installation of the software-based artwork, but its change does usually not cause malfunction.

Returning to the discussion of sustainability of preservation measures it can be said, the more external dependencies can be eliminated, the more sustainable the

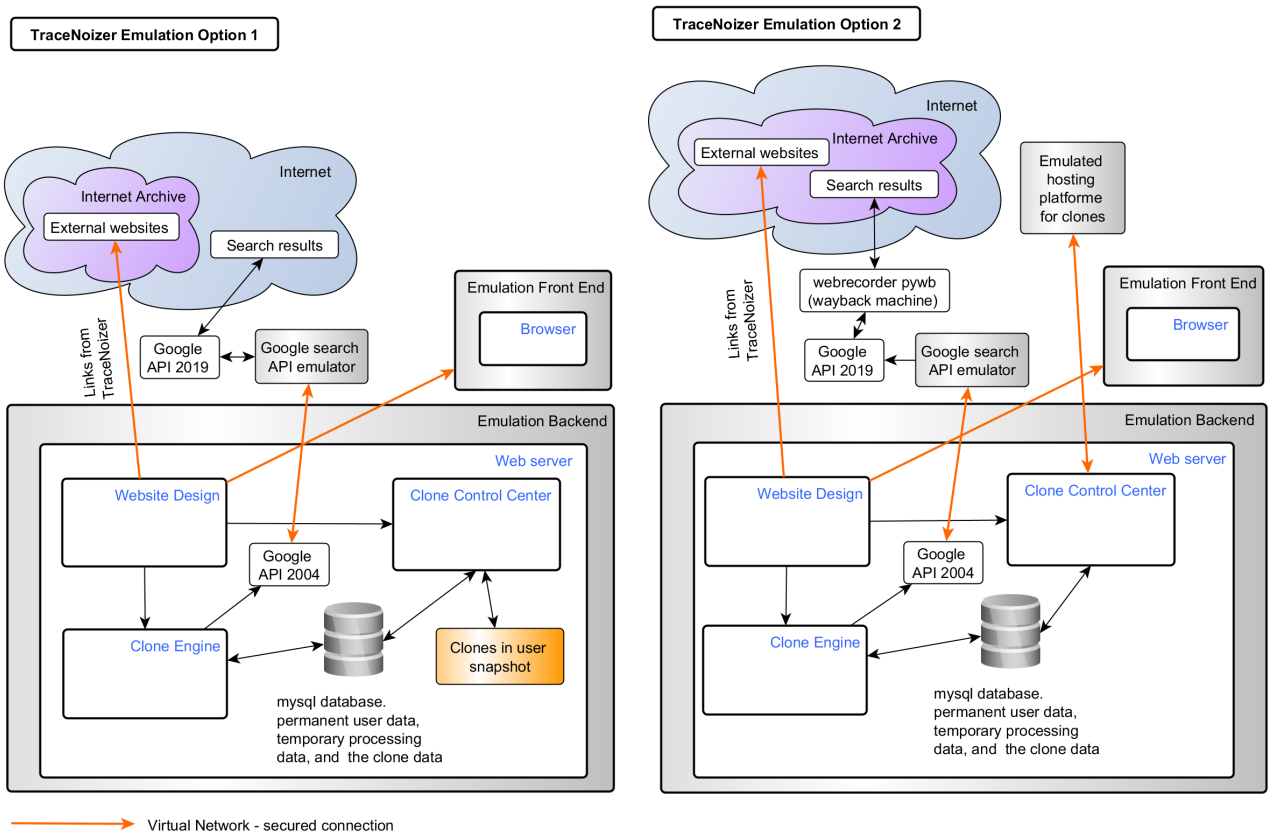

Figure 4: Emulation strategy options for *TraceNoizer*, showing different degrees of emulation. In Option 1, the emulated *TraceNoizer* interacts with the current live Web and generated clones are not exposed to the live Internet. In Option 2, Internet Archive's Wayback Machine is used to let *TraceNoizer* interact with an archived version of the Web and generated clones are exposted to the live Internet.

preservation strategy is. From that point of view, the simulation strategy is the best strategy, as it replaces the evolving Web environment with a stabilized archive. Besides the reduction of dependencies, the reduction of maintenance and security aspects are dominating in the assessment of sustainability of preservation strategies for internet-based artworks. As can be seen with TraceNoizer, the migrated website is unsafe and prone to web attacks. In contrast, it can be very economical to restart an emulation regularly instead of maintaining the server that can quickly take an hour or more weekly. The isolation of several internet-based artworks with different needs and dependencies from each other can be another argument for emulation / virtualization.

In the future, a good compromise between reduction of maintenance, security aspects and functionality will be crucial for the preservation of internet-based artworks.

IX. Acknowledgment

This research was funded as part of NACCA (New Approaches in the Conservation of Contemporary Art, www.nacca.eu), a Marie Skłodowska-Curie Innovative Training Network of the European Union. The House of Electronic Arts provided the case study *TraceNoizer* (2001), LAN (the artists) provided information and an interview about the work and Fabian Thommen, member of LAN migrated *TraceNoizer*. Rafael Gieschke and Klaus Rechert, University of Freiburg carried out the emulation of TraceNoizer.

References

[1] A. Dekker, "Enabling the future, or how to survive forever: A study of networks, processes and ambiguity in net art and the need for an expanded practice of conservation," Doctoral Thesis, Goldsmiths, University of London, London, 2014.

[2] J. Phillips, D. Engels, E. Dickson, and J. Farbowitz, *Restoring brandon. shu lea cheang's early web artwork*, www.guggenheim.org, 2017. [Online]. Available: `https://www.guggenheim.org/blogs/checklist/restoring-brandon-shu-lea-cheangs-early-web-artwork` (visited on 07/01/2017).

[3] J. Phillips, D. Engel, J. Farbowitz, and K. T. Rosenberg, *The guggenheim restores john f. simon jr.'s early web artwork "unfolding object"*, 2018. [Online]. Available: `https://www.guggenheim.org/blogs/checklist/the-guggenheim-restores-john-f-simon-jr-early-web-artwork-unfolding-object` (visited on 11/26/2018).

[4] T. Miksa, R. Mayer, and A. Rauber, "Ensuring sustainability of web services dependent processes,"

International Journal of Computational Science and Engineering, vol. 10, no. 1/2, p. 70, 2015, ISSN: 1742-7185. DOI: 10.1504/IJCSE.2015.067058.

[5] D. Espenschied and K. Rechert, "Fencing apparently infinite objects: Defining productive object boundaries for performative digital objects," in *iPres 2018*, iPres 2018, Ed., 2018.

[6] M. Kelly, *Browser-based digital preservation*, 2014/07/07. [Online]. Available: https://www.slideshare.net/matkelly01/browserbased-digital-preservation.

[7] M. McKeehan and I. Kreymer, *Symmetrical web archiving with webrecorder, a browser-based tool for digital social memory. an interview with ilya kreymer*, The National Digital Stewardship Residency New York, 2016/02/23.

[8] C. Roeck, K. Rechert, and J. Noordegraaf, "Evaluation of preservation strategies for an interactive, software-based artwork with complex behavior using the case study horizons (2008) by geert mul.," in *iPres 2018*, iPres 2018, Ed., 2018. [Online]. Available: https://osf.io/y3gcu/.

[9] *How google search works: Useful responses take many forms*, 2019. [Online]. Available: https://www.google.com/search/howsearchworks/responses/#?modal_active=none.

[10] R. Davoli, "Vde: Virtual distributed ethernet," in *First International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities*, IEEE, 2005, pp. 213–220.

[11] M. Kruisselbrink, J. Song, J. Archibald, and A. Russell, "Service workers 1," W3C, W3C Working Draft, Nov. 2017, https://www.w3.org/TR/2017/WD-service-workers-1-20171102/.